

Interactive Computing and Processing of NASA Land Surface Observations using Google Earth Engine

Andrew Molthan¹, Jason Burks¹, and Jordan Bell²

¹Earth Science Office / NASA Marshall Space Flight Center, Huntsville, AL

²University of Alabama in Huntsville, Huntsville, Alabama

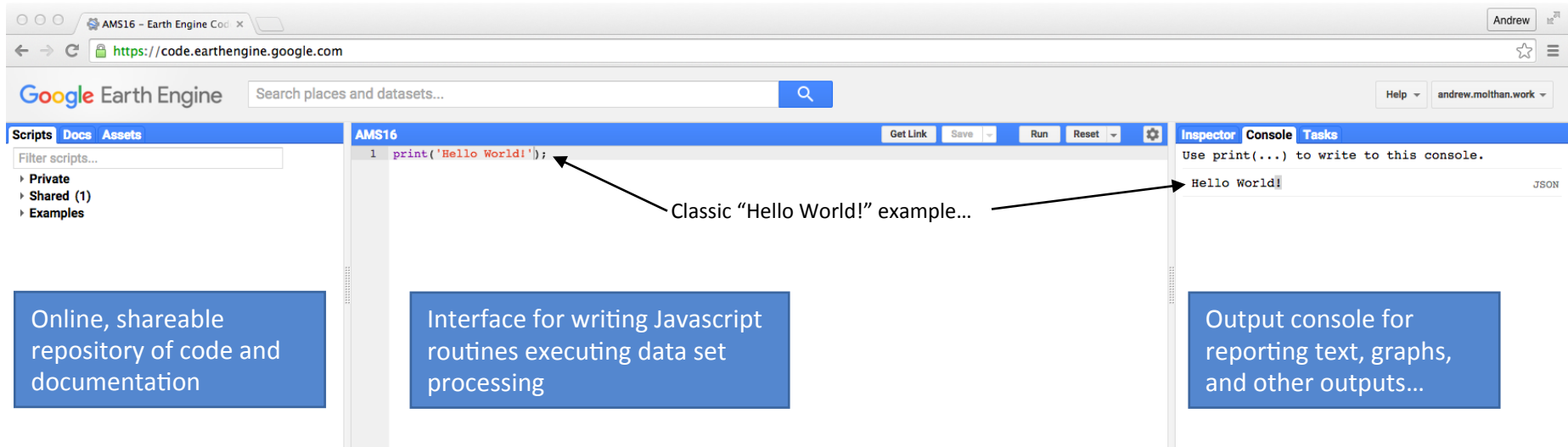
andrew.molthan@nasa.gov

Presentation 8.2, 32nd Conference on Environmental Information Processing Technologies (EIPT)
2016 (96th) AMS Annual Meeting, New Orleans, LA

Background

- Google's Earth Engine offers a “big data” approach to processing large volumes of NASA and other remote sensing products. <https://earthengine.google.com/>
- Interfaces include a Javascript or Python-based API, useful for accessing and processing over large periods of record for Landsat and MODIS observations.
 - Other data sets are frequently added, including weather and climate model data sets, etc.
- Demonstrations here focus on exploratory efforts to perform land surface change detection related to severe weather, and other disaster events.

Javascript API (<http://ee-api.appspot.com>)



The screenshot displays the Google Earth Engine Code Editor interface. The browser address bar shows <https://code.earthengine.google.com>. The page title is "AMS16 - Earth Engine Code Editor". The main interface is divided into several panels:

- Scripts Panel:** Located on the left, it shows a list of scripts under the heading "Filter scripts...". The list includes "Private", "Shared (1)", and "Examples".
- Code Editor:** The central panel shows a script named "AMS16" with the following code:

```
1 print('Hello World!');
```
- Inspector Panel:** Located on the right, it shows the execution results of the script. It displays "Hello World!" and "JSON".
- Console Panel:** Also on the right, it shows the output of the script, which is "Hello World!".

Annotations and callouts highlight key features:

- A blue box on the left states: "Online, shareable repository of code and documentation".
- A blue box in the center states: "Interface for writing Javascript routines executing data set processing".
- A blue box on the right states: "Output console for reporting text, graphs, and other outputs...".
- An arrow points from the text "Classic 'Hello World!' example..." to the code in the editor and the output in the console.



Javascript API (http://ee-api.appspot.com)

The screenshot displays the Google Earth Engine web interface. The top navigation bar includes the Google Earth Engine logo, a search bar with the text 'modis ndwi', and a user profile 'Andrew'. Below the navigation bar, the interface is divided into three main sections: Scripts, Code Editor, and Inspector/Console.

Scripts Panel: A list of scripts is shown, including 'Private', 'Shared (1)', and 'Examples'.

Code Editor: A script titled 'AMS16' is displayed. The script is as follows:

```
1 // Create a 'collection', in this case Aqua MODIS Normalized Difference Water Index (NDWI)
2 var collection = ee.ImageCollection('MODIS/MYD09GA_NDWI');
3 // Filter that collection to the period of June 1, 2010 through August 31, 2010
4 var aqua_ndwi_summer2010 = collection.filterDate('2010-06-01','2010-08-31');
5 // Print the contents to the console
6 print(aqua_ndwi_summer2010);
7
```

Inspector/Console Panel: The console shows the output of the script, indicating that the collection 'MODIS/MYD09GA_NDWI' has been successfully retrieved and filtered. The output is a JSON object with the following structure:

```
{
  "type": "ImageCollection",
  "id": "MODIS/MYD09GA_NDWI",
  "version": 1449781080979000,
  "bands": List (1 element),
  "features": List (91 elements)
}
```

Map: A map of the central United States is displayed, showing major cities and highways. The map is centered on the area around Lincoln, Nebraska, and Kansas City, Missouri. The map data is from 2016, and the scale is 20 km.

Simple Example:

1. Retrieve Aqua MODIS NDWI for all scenes in Summer (JJA) 2010.
2. Confirm retrieval by printing information to the console.

Javascript API (http://ee-api.appspot.com)

AMS16 - Earth Engine Code Editor

https://code.earthengine.google.com

Google Earth Engine modis ndwi

Help andrew.molthan.work

Scripts Docs Assets

Inspector Console Tasks

AMS16

Adding:

1. Get the same product and period for 2011.

2. Retain the maximum.

3. Difference 2011-2010

4. Display

5. Compare layers

1 // Create a 'collection', in this case Aqua MODIS Normalized Difference Water Index (NDWI)

2 var collection = ee.ImageCollection('MODIS/MYD09GA_NDWI');

3 // Filter that collection to the period of June 1, 2010 through August 31, 2010

4 var aqua_ndwi_summer2010 = collection.filterDate('2010-06-01','2010-08-31');

5 // Create another period for Summer 2011.

6 var aqua_ndwi_summer2011 = collection.filterDate('2011-06-01','2011-08-31');

7 // Create a single image to display -- the maximum for each period.

8 var aqua_ndwi_summer2010_max = aqua_ndwi_summer2010.max();

9 var aqua_ndwi_summer2011_max = aqua_ndwi_summer2011.max();

10 // Difference these parameters

11 var diff = aqua_ndwi_summer2011_max.subtract(aqua_ndwi_summer2010_max);

12

13 // Display on a map.

14 var ndwiViz = {min:0, max:1}; // Grayscale visualization for NDWI

15 var diffViz = {min:-0.2, max:0.2}; // Grayscale visualization for difference

16 Map.addLayer(aqua_ndwi_summer2010_max, ndwiViz, 'Aqua MODIS Max NDWI Summer 2010');

17 Map.addLayer(aqua_ndwi_summer2011_max, ndwiViz, 'Aqua MODIS Max NDWI Summer 2011');

18 Map.addLayer(diff, diffViz, 'Difference NDWI Maximum');

19 centerMap(-95,40,8); // Center display on the Nebraska/Iowa/Missouri borders

Use print(...) to write to this console.

Layers

Map Satellite

Flooded Missouri River

NDWI Maximum Summer 2011

Data downloaded: None
Processing Time: ~1-2 min.

Map data ©2016 Google 20 km Terms of Use Report a map error

Javascript API (http://ee-api.appspot.com)

AMS16 - Earth Engine Code Editor

https://code.earthengine.google.com

Google Earth Engine modis ndwi

Help andrew.molthan.work

Scripts Docs Assets

Adding:

1. Get the same product and period for 2011.
2. Retain the maximum.
3. Difference 2011-2010
4. Display
5. Compare layers

AMS16

Get Link Save Run Reset

```
1 // Create a 'collection', in this case Aqua MODIS Normalized Difference Water Index (NDWI)
2 var collection = ee.ImageCollection('MODIS/MYD09GA_NDWI');
3 // Filter that collection to the period of June 1, 2010 through August 31, 2010
4 var aqua_ndwi_summer2010 = collection.filterDate('2010-06-01', '2010-08-31');
5 // Create another period for Summer 2011.
6 var aqua_ndwi_summer2011 = collection.filterDate('2011-06-01', '2011-08-31');
7 // Create a single image to display -- the maximum for each period.
8 var aqua_ndwi_summer2010_max = aqua_ndwi_summer2010.max();
9 var aqua_ndwi_summer2011_max = aqua_ndwi_summer2011.max();
10 // Difference these parameters
11 var diff = aqua_ndwi_summer2011_max.subtract(aqua_ndwi_summer2010_max);
12
13 // Display on a map.
14 var ndwiViz = {min:0, max:1}; // Grayscale visualization for NDWI
15 var diffViz = {min:-0.2, max:0.2}; // Grayscale visualization for difference
16 Map.addLayer(aqua_ndwi_summer2010_max, ndwiViz, 'Aqua MODIS Max NDWI Summer 2010');
17 Map.addLayer(aqua_ndwi_summer2011_max, ndwiViz, 'Aqua MODIS Max NDWI Summer 2011');
18 Map.addLayer(diff, diffViz, 'Difference NDWI Maximum');
19 centerMap(-95,40,8); // Center display on the Nebraska/Iowa/Missouri borders
```

Inspector Console Tasks

Use print(...) to write to this console.

Layers Map Satellite


Flooded Missouri River

Difference of NDWI Maximums for Summer 2011-2010

Data downloaded: None
Processing Time: ~1-2 min.

Map data ©2016 Google 20 km Terms of Use Report a map error

Python API (https://developers.google.com/earth-engine/python_install?hl=en)

 Google Developers

Search

andrew.molthan.work@gmail.com
Sign out

Products > Google Earth Engine API

Google Earth Engine API

GUIDES REFERENCE TUTORIALS

SEND FEEDBACK

Developer's Guide

Introduction

Get Started

Earth Engine Code Editor

[Python Installation](#)

Image

Image Overview

Image Visualization

Image Information and Metadata

Creating Images

Mathematical Operations

Relational, Conditional and Boolean Operations

Convolutions

Morphological Operations

Gradients

Edge Detection

Spectral Transformations

Texture

Object-based Methods

ImageCollection

ImageCollection Information and Metadata

Creating an ImageCollection

Filtering an ImageCollection

Mapping over an ImageCollection

Reducing an ImageCollection

Compositing and Mosaicking

Iterating over an ImageCollection

Geometry, Feature, FeatureCollection

Geometry Overview

Python installation



The Earth Engine Python API is distributed as a Python package that is [hosted on Github](#). The following instructions give an overview of installing the Google Earth Engine Python API. To use the Earth Engine Python API you'll need to [install the client library and its dependencies](#) on your computer and then [set up authentication credentials](#).

Installing the client library

Ubuntu Linux & Mac OS X installation

After the initial set up, the installation flows for Mac OS X and Ubuntu are nearly identical.

1. Set up pip and Python

[PIP](#) is a package manager for Python. The following installation instructions assume that you are using it.

Ubuntu Linux

Verify that you have Python 2.6 or 2.7:

```
python --version
```

If needed, install 2.6 or 2.7 with `apt-get`. Then pip can be installed with:

```
sudo apt-get install python-pip
```

Mac OS X

The installation instructions assume that you are using Mac OS X 10.9+, the [Homebrew](#) Mac OS package manager, and the [pip](#) Python package manager. Feel free to use a different package manager such as [Fink](#) or [MacPorts](#) if you prefer.

Contents

Installing the client library

Ubuntu Linux & Mac OS X installation

Windows Installation

Uninstalling the Library

Uninstalling manually

Setting Up Authentication Credentials

Testing the installation

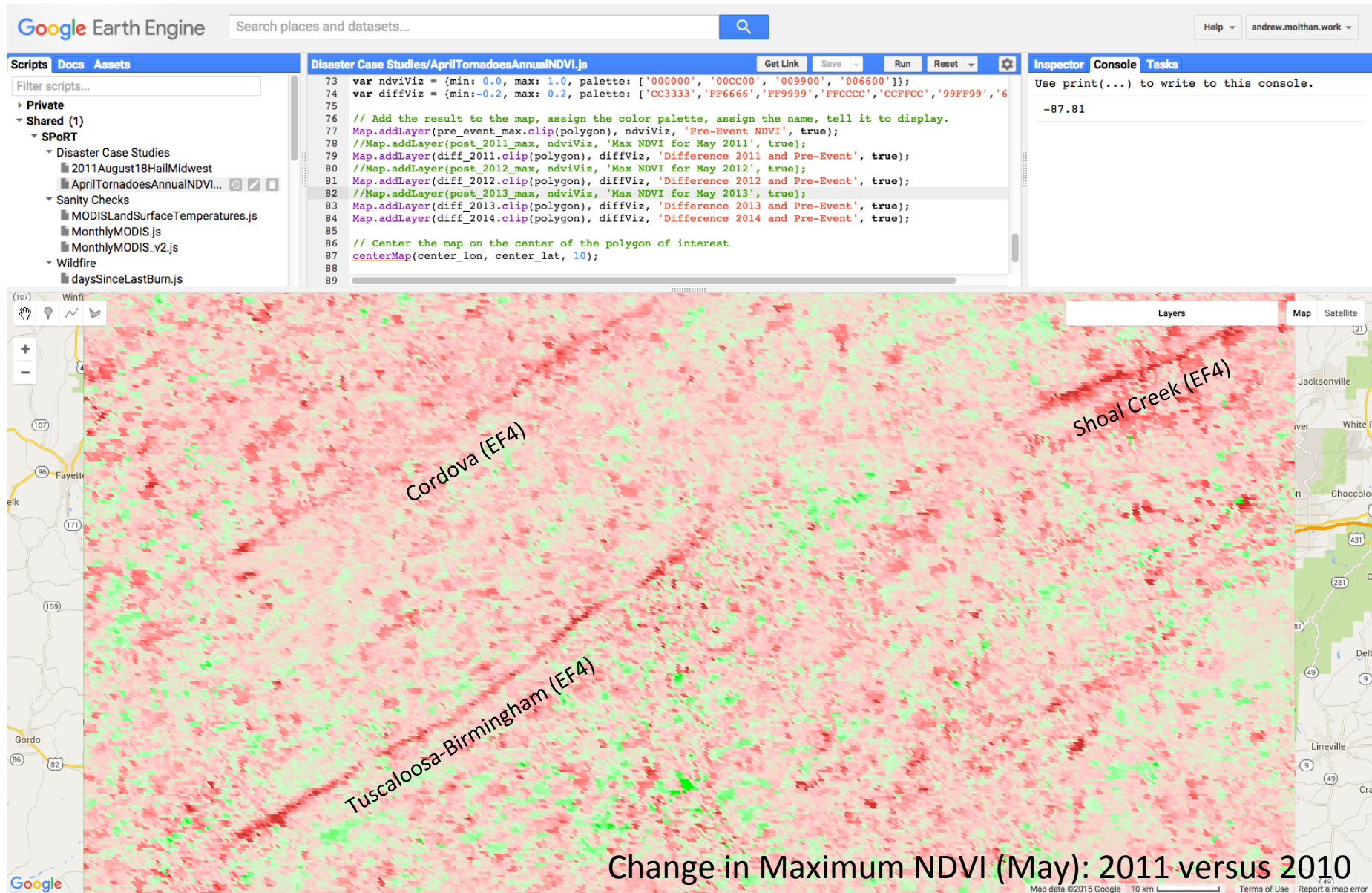
Coding in the Python API

Python packages are available for the API as well.

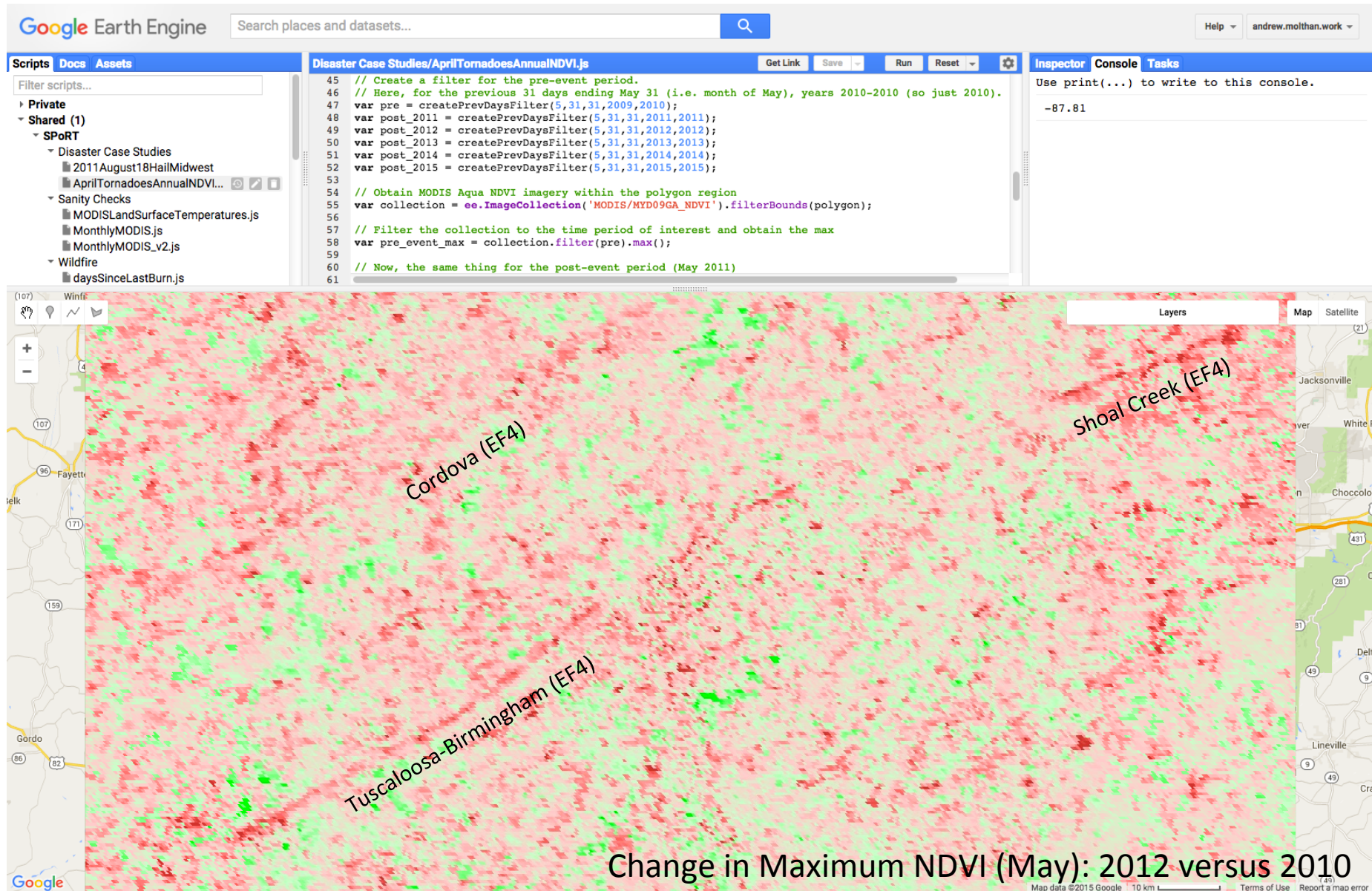
Allows users to authenticate their interaction with Earth Engine and write scripts, etc. to perform processing and other Python manipulations.

Preferable in some ways to using a web client for interaction and coding.

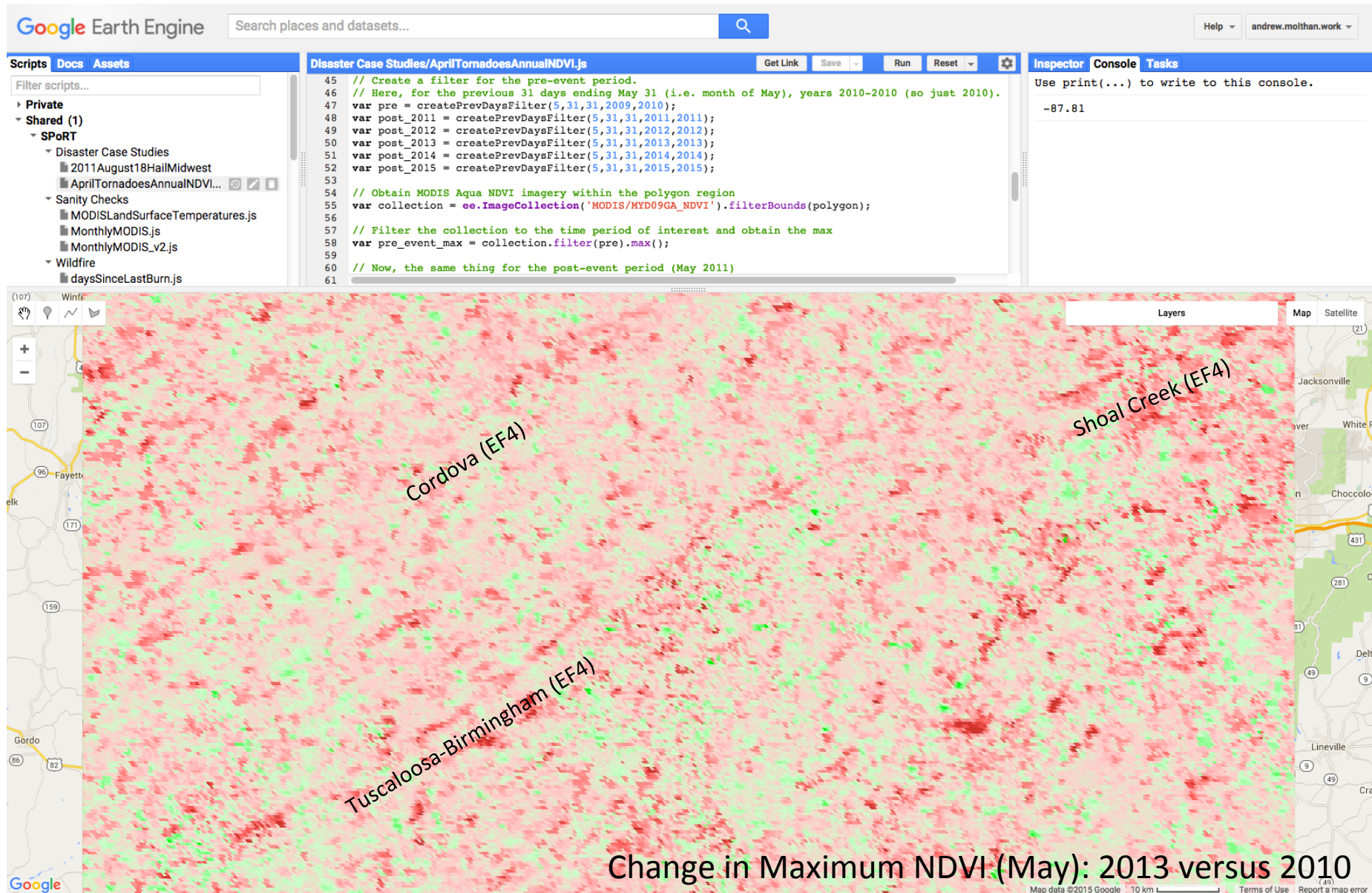
Applications: Land Surface Change



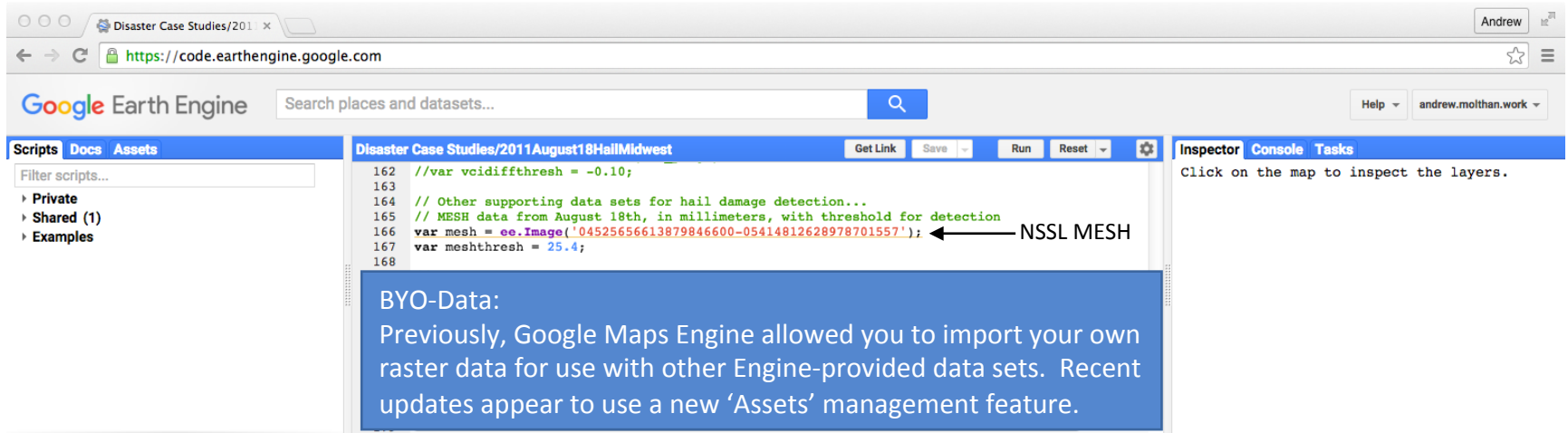
Applications: Land Surface Change



Applications: Land Surface Change



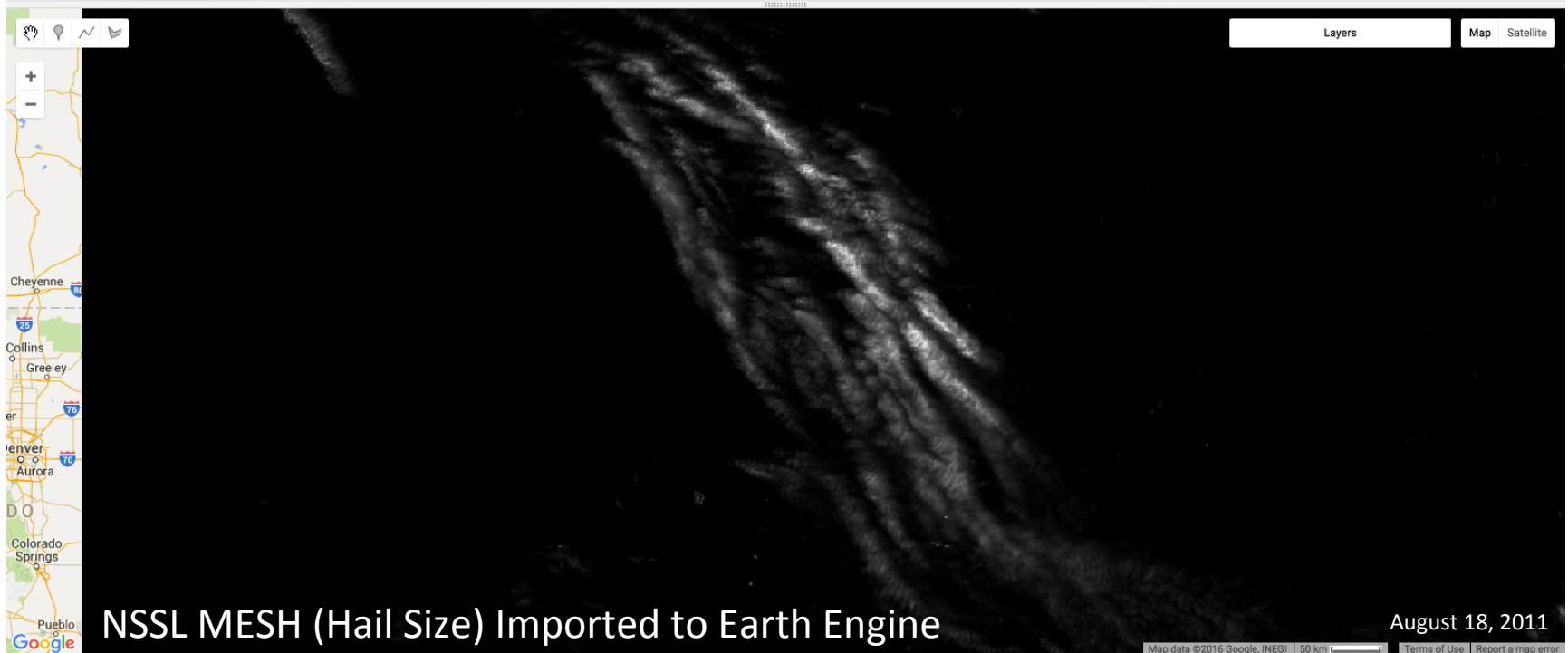
Applications: Hail Damage



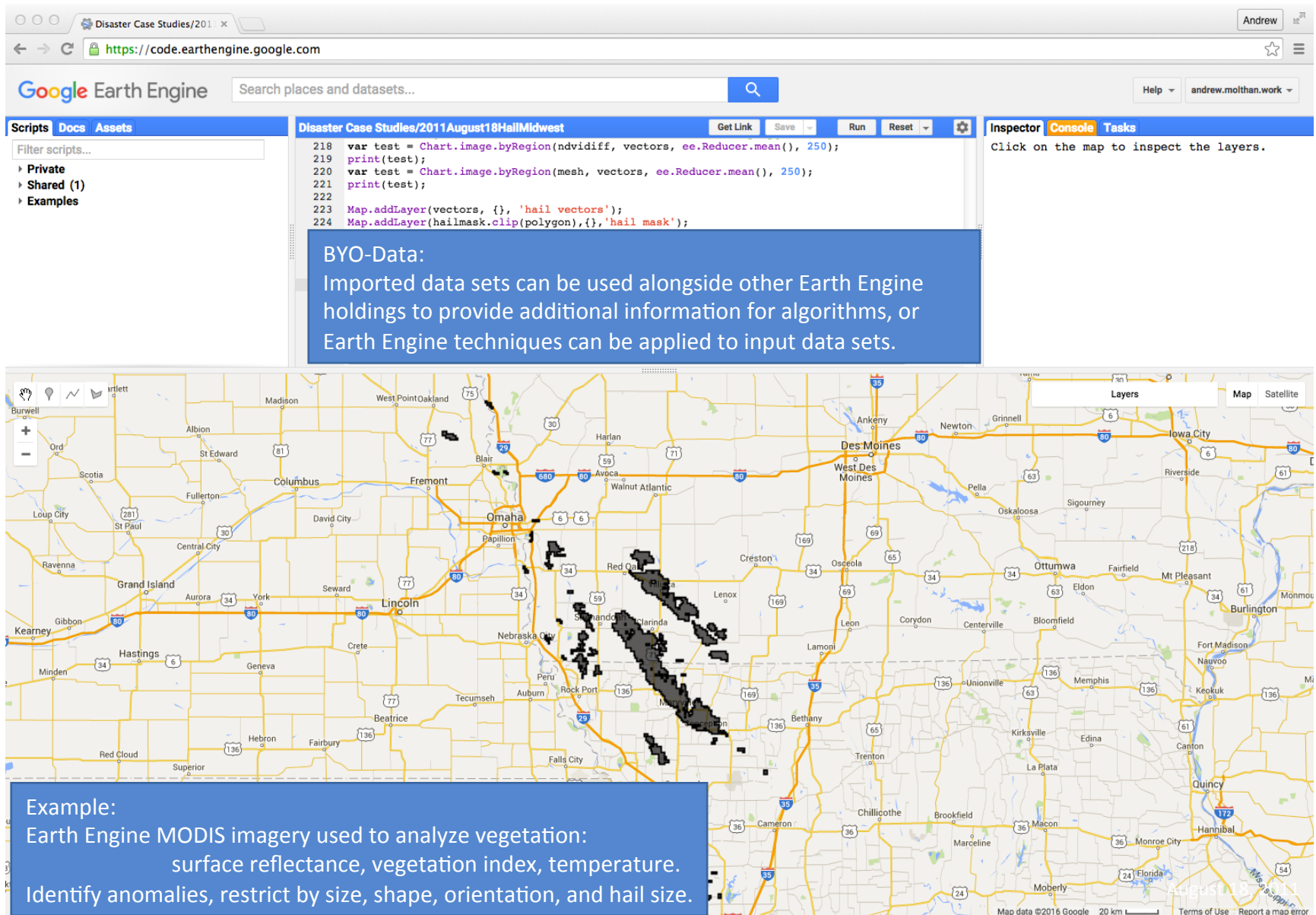
The screenshot shows the Google Earth Engine web interface. The browser address bar displays `https://code.earthengine.google.com`. The left sidebar contains the 'Scripts' tab with a list of scripts, including 'Disaster Case Studies/2011August18HailMidwest'. The main editor area shows a JavaScript script for importing and processing NSSL MESH data. A blue callout box highlights the 'BYO-Data' section of the script.

```
162 //var voidifftresh = -0.10;
163
164 // Other supporting data sets for hail damage detection...
165 // MESH data from August 18th, in millimeters, with threshold for detection
166 var mesh = ee.Image('04525656613879846600-05414812628978701557');
167 var meshthresh = 25.4;
168
```

BYO-Data:
Previously, Google Maps Engine allowed you to import your own raster data for use with other Engine-provided data sets. Recent updates appear to use a new 'Assets' management feature.



Applications: Hail Damage



The screenshot displays the Google Earth Engine web interface. The top navigation bar includes the Google Earth Engine logo, a search bar, and user information. The left sidebar shows a list of scripts under 'Disaster Case Studies/2011August18HailMidwest'. The main panel shows a JavaScript script for analyzing hail damage. The right sidebar shows the 'Inspector' and 'Console' tabs. A map of the central United States is displayed at the bottom, with a dark shaded area representing hail damage. A blue text box is overlaid on the map, providing an example of how Earth Engine MODIS imagery can be used to analyze vegetation and identify anomalies.

```
218 var test = Chart.image.byRegion(ndvidiff, vectors, ee.Reducer.mean(), 250);
219 print(test);
220 var test = Chart.image.byRegion(mesh, vectors, ee.Reducer.mean(), 250);
221 print(test);
222
223 Map.addLayer(vectors, {}, 'hail vectors');
224 Map.addLayer(hailmask.clip(polygon), {}, 'hail mask');
```

BYO-Data:
Imported data sets can be used alongside other Earth Engine holdings to provide additional information for algorithms, or Earth Engine techniques can be applied to input data sets.

Example:
Earth Engine MODIS imagery used to analyze vegetation:
surface reflectance, vegetation index, temperature.
Identify anomalies, restrict by size, shape, orientation, and hail size.

Strengths / Weaknesses

- Strengths

- Earth Engine data holdings are tremendous, and the streamlined Javascript/Python API is capable of processing large volumes of data very quickly.
- Relatively simple, big data platform for exploring various remote sensing products and capabilities.
- Numerous and increasing number of high volume data sets available that could be difficult to download and process locally.

- Weaknesses

- Primarily focused on raster types of data – large gridded or swath products of comparable resolution.
- Most products still require some understanding of the raw data – many appear to be a literal depositing of the source data with no additional post-processing.
 - Ex: Temperatures in scaled units, not K
- Documentation could benefit from a few use examples, and some experiences where documentation or errors were cryptic.

Questions?

- Check out some related talks!
 - Near-Real Time Severe Weather Damage Identification Algorithm for Vegetation: Development and Early Results
 - J. Bell, University of Alabama in Huntsville
 - Thursday, 3:30 pm, 252
- andrew.molthan@nasa.gov